THE FRACTAL OF BRANCHES
树枝的分形

参赛队员:李明哲 梁栋 王佳瑞

带队老师:刘正峰

辅导老师:吕栋亮

摘要：本文研究生活中一些常见树枝的分形，鉴于分形图形由生成元决定，因此对树枝分形的研究归结为对生成元的研究。本文考虑两类生成元，并且根据生成元的节点序列是否递减和末端是否细分把每一类生成元分成四种。借助于计算机程序，本文给出了每一种分形的三维图形。
关键词：树枝，分形，生成元，线段序列，节点序列。

Abstract

This paper mainly focuses on the fractal of branch which can be seen in the life. It is known that the fractal is completely determined by the generator,thus the study of fractal comes down to the study of its generator. This paper considers two types of generators,and each type is divided into four kinds. With computer programs, this paper presents each kind of fractal graphics.

Key words and phrases:branch,fractal,generator,the sequence of line segment,node sequence.

目录

1 引言 **3**

2 正文 **4**

2.1 第一类生成元 4

2.2 第二类生成元 7

3 附录 **9**

1 引言

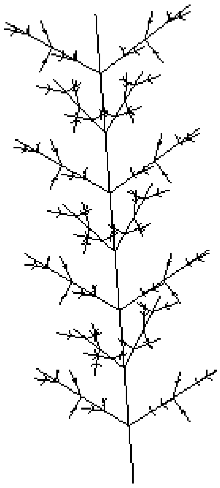

图 1: 迎春花枝条的分形

本文我们主要研究树的分形，它开始引起我们的注意是在去年的夏天，那时我们在溪边的树下乘凉，旁边的一颗不知名的花(*后来才知道那就是迎春花*) 的枝条长得是那样的规则，它的主干在每个节点上生出两个枝条，这两个枝条关于主干对称，而且与主干在一个平面上，这样在每个节点上都确定了一个平面，更有意思的是在交替的结点上，所确定的两个平面是相互垂直的，仔细观察主枝上生长出的小枝条时，却发现小枝条仍按照主枝条的生长方式伸出更细小的枝条，这一现象使我们惊奇不已，后来我们用$Mathematica$程序(程序部分参考了≪*数学实验*≫ [2])模拟了它的图形，如图 1。

生活中会发现许多类似的例子，一朵云彩的每一个小的部分与整体的形状很相似，一棵树长出的树枝与树的主干形状大致相同，你也许多少会注意到地图上的海岸线的每一个小部分也与整体相似，其实这些源自生活中的例子，在数学上已有一个学科来描述它，那就是分形($fractal$)。

分形是空间上一些点的集合，并且基本具有下面经典的几何性质：

$*$ 分形集都具有任意小尺度下的比例细节，即具有无限精细结构；

$*$ 分形集无法用传统几何语言来描述，它不是某些简单方程的解集，也不是满足某些条件的点的轨迹；

∗ 分形集具有某种自相似形式，包括近似和统计上的自相似；
∗ 分形集一般可以用简单的方法定义和产生，如迭代；
∗ 按某种维数定义，分形集的分形维数大于相应的拓扑维数。

注：针对不同图形，有时它可能只具有上面大部分性质，而不满足某个性质，但我们一般仍然把它归入分形。这个定义引自潘金贵的≪分形艺术程序设计≫ [1]。

有了上面的定义，我们可以发现前面所述的迎春花枝条的形状是分形，后来我们还发现许多种树的枝条也有上述的生长规律，因此在本文中我们多用树枝的分形来描述它们。

2 正文

分形的基本特征完全由生成元决定。给定一个生成元，我们就可以生成各种各样的分形图形。因此研究树的分形归结为对不同生成元的研究。由于篇幅有限，本文我们主要研究两类生成元：一类是图 2所示的生成元(见 2.1，第 4页)，另一类是图 11所示的生成元(见 2.2，第 7页)

2.1 第一类生成元

这节我们讨论的生成元满足如下的性质：树的主干被分成均匀的n段，这样会相应的得到$n-1$个节点，每个节点生出两个枝条，这两个枝条关于主干对称，且与主干在同一平面上，伴随每个节点都有一个平面，并且在相邻的节点上两个平面垂直(如图 2)。

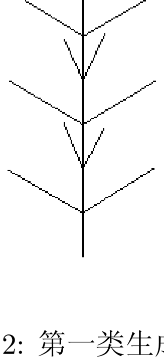

图 2: 第一类生成元

由于主干被均匀的分成了n段，可按照第n段是否分出更小的枝条来分类，在迭代的过程中我们还要考虑下一个生成元的节点数是否与上一层生成元的节点数相同，我们观察真实的树枝时，发现小枝上的节点数通常比它的主枝上的节点数要少，因此这种情况我们应该考虑在内。若假设第k次迭代的生成元等分的线段数为X_k，节点数为J_k，则有$X_k = J_k + 1$，而且按照刚才的讨论，节点序列$\{J_k\}$可为两类: (a) $J_1 = \cdots = J_k = \cdots$; (b) $J_1 > \cdots > J_k > \cdots$.

因此，对第一种生成元，在迭代时我们分成四种情况加以讨论：

*(1)*主枝的最后一段细分且节点序列$\{J_k\}$为情况(a)*:*

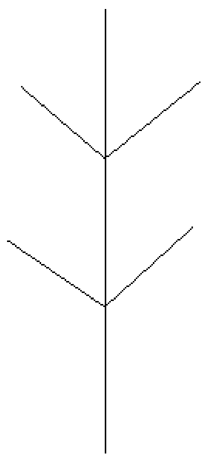

图 3:

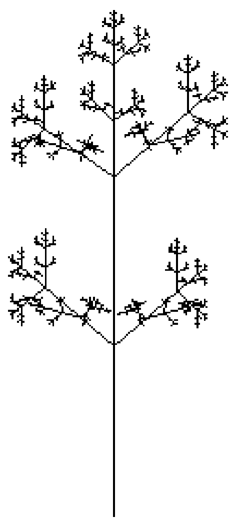

图 4:

设$n = 3$,则对任意的k，$J_k = 2$，从而序列$\{J_k\}$满足$J_1 = \cdots = J_k = \cdots$，生成元如图 3，我们让生成元进行迭代可以得到图 4。

*(2)*主枝的最后一段细分且节点序列$\{J_k\}$为情况(b)*:*

我们讨论节点序列$\{J_k\}$的一种特殊情况，假定$X_{k+1} = X_k - 1$，由于$X_k = J_k+1$，从而$J_{k+1} = J_k-1$，这时序列$\{J_k\}$满足 $J_1 > \cdots > J_k > \cdots$。

设$n = 5$，生成元如图 5，我们让生成元进行迭代可以得到图 6。

*(3)*主枝的最后一段不细分且节点序列$\{J_k\}$为情况(a)*:*

设$n = 4$,我们让生成元（如图 7）进行迭代可以得到图 8。

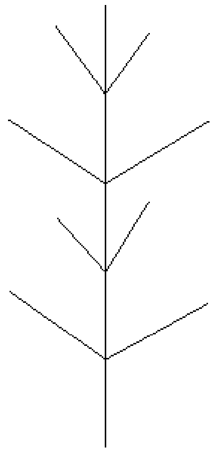

图 5:

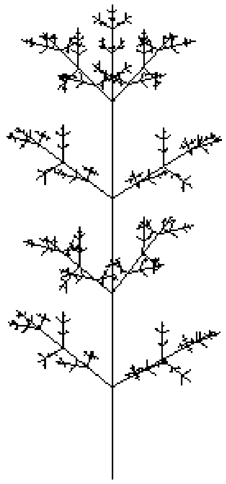

图 6:

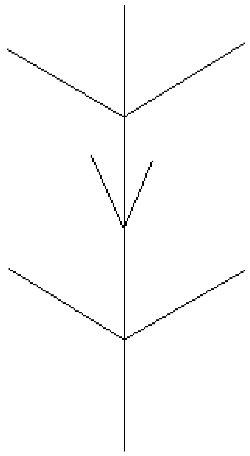

图 7:

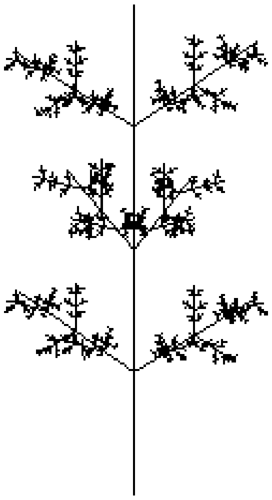

图 8:

*(4)*主枝的最后一段不细分且节点序列$\{J_k\}$为情况(b):

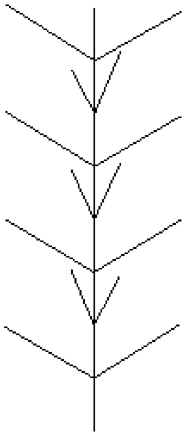

图 9:

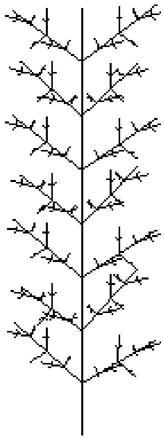

图 10:

由于满足 $J_1 > \cdots > J_k > \cdots$ 的节点序列$\{J_k\}$ 可以有许多种情况，这里我们选取比较特殊的一种，即假定$X_{k+1} = \lfloor X_k/2 \rfloor$（其中$\lfloor\ \rfloor$表示一个实数的最大整数部分，简称为对一个实数向下取整）。由于$X_k = J_k + 1$，从而有$J_{k+1} + 1 = \lfloor (J_k + 1)/2 \rfloor$，这时$\{J_k\}$满足 $J_1 > \cdots > J_k > \cdots$ 。

设$n = 8$,我们让生成元（如图 9）进行迭代可以得到图 10。

2.2 第二类生成元

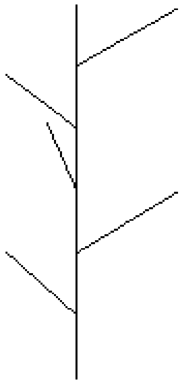

图 11: 第二类生成元

这节我们讨论的生成元满足如下的性质：树的主干被分成均匀的n段，这样会相应的得到$n-1$个节点，每个节点生出一个枝条，生成元上相邻的两个枝条投影到水平面上相差一个固定的角度，这里假定为$\frac{2\pi}{3}$（如图 11）。

类似第 2.1节的讨论，这里第二种生成元仍按照第n段是否细分以及节点是否递减来分，共四种。

*(1)*第n段细分且节点J_k恒为常值$n-1$

设$n=6$，生成元如图 11，生成元迭代后得到图 12。

*(2)*第n段细分且节点序列$\{J_k\}$是递减序列

令$X_{k+1} = X_k - 1$，由于$X_k = J_k + 1$，从而$J_{k+1} = J_k - 1$，这时序列$\{J_k\}$满足 $J_1 > \cdots > J_k > \cdots$。

设$n=6$,生成元如图 11，生成元迭代后得到图 13。

*(3)*第n段不细分且节点J_k恒为常值$n-1$

设$n=6$，生成元如图 11，生成元迭代后得到图 14。

*(4)*第n段不细分且节点序列$\{J_k\}$是递减序列

这里我们令节点序列$\{J_k\}$与*(2)*中的相同，即$J_{k+1} = J_k - 1$，这时序列$\{J_k\}$满足 $J_1 > \cdots > J_k > \cdots$。

设$n=6$,生成元如图 11，生成元迭代后得到图 15。

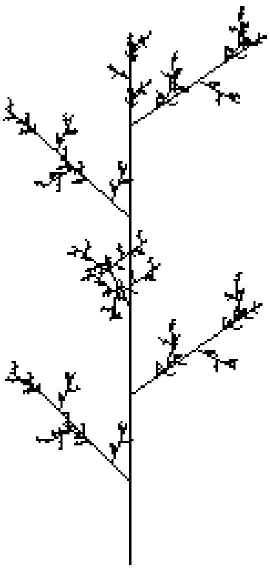

图 12:

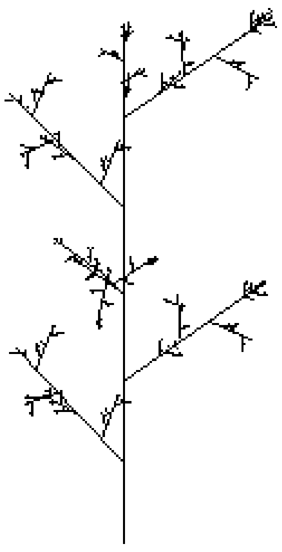

图 13:

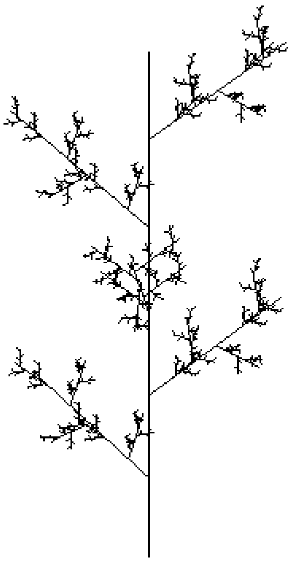

图 14:

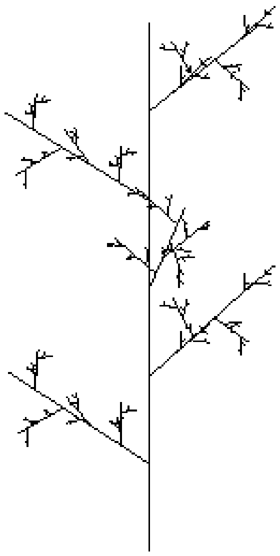

图 15:

3 附录

我们用$Mathematica$程序来实现树枝在分形状态下的图形，为节省篇幅，仅给出第一类生成元中的第二种（$Program1$）和第二类生成元中的第一种（$Program2$）程序如下：

Program1 :

$oy = \{\{0,0,0\},\{0,1,0\}\};$

$oz = \{\{0,0,0\},\{0,0,1\}\};$

$LineSet = \{\{\{0,0,0\},\{0,0,1\}\}\};$

$LineCounting = \{1\};$

$n = 5;$

$dividenumber = 3.5;$

$\theta = \pi/3;$

$f[a_, b_, \eta_] := Cos[\eta] * a + Sin[\eta] * b;$

$LineOnPlane[a_, b_, \eta_] := Block[\{temp, vector, vector1, vector2\},$

$\quad vector1 = a[[2]] - a[[1]];$

$vector2 = b[[2]] - b[[1]];$

$vector = Cross[vector1, vector2];$

$vector2 = Cross[vector, vector1];$

$vector2 = Normalize[vector2];$

$temp = a[[1]] + f[vector1, vector2, \eta];$

$\{a[[1]], temp\}];$

$LineOffPlane[a_, b_, \eta_] := Block[\{temp, vector1, vector2\},$

$vector1 = a[[2]] - a[[1]];$

$vector2 = b[[2]] - b[[1]];$

$vector2 = Cross[vector1, vector2];$

$vector2 = Normalize[vector2];$

$temp = a[[1]] + f[vector1, vector2, \eta];$

$\{a[[1]], temp\}];$

$LineNormalize[a_] := Block[\{temp\},$

$temp = a[[1]] + Normalize[a[[2]] - a[[1]]];$

$\{a[[1]], temp\}];$

$LineScaling[a_, t_] := Block[\{temp\},$

$temp = a[[1]] + t * (a[[2]] - a[[1]]);$

$\{a[[1]], temp\}];$

$BranchGenerate[a_, b_, number_] :=$

$Block[\{i, length, vector, temp, temp1, temp2, tempa, tempb\},$

$AppendTo[LineSet, \{a[[1]]+(number-1)*(a[[2]]-a[[1]])/number, a[[2]]\}];$

$length = Norm[a[[2]] - a[[1]]];$

$For[i = 1, i < number,$

$vector = i * (a[[2]] - a[[1]])/number;$

$temp1 = \{a[[1]] + vector, a[[2]]\};$

$temp2 = \{a[[1]] + vector, a[[1]] + vector + b[[2]] - b[[1]]\};$

$temp1 = LineNormalize[temp1];$

$If[Mod[i, 2] == 1,$

$temp = LineOffPlane[temp1, temp2, \theta];$

$temp = LineNormalize[temp];$

$AppendTo[LineSet, LineScaling[temp, length/dividenumber]];$

$temp = LineOffPlane[temp1, temp2, -\theta];$

$temp = LineNormalize[temp];$

$AppendTo[LineSet, LineScaling[temp, length/dividenumber]],$

$temp = LineOnPlane[temp1, temp2, \theta];$

$temp = LineNormalize[temp];$

$AppendTo[LineSet, LineScaling[temp, length/dividenumber]];$

$temp = LineOnPlane[temp1, temp2, -\theta];$

$temp = LineNormalize[temp];$

$AppendTo[LineSet, LineScaling[temp, length/dividenumber]]$

$];$

$i++];$

$0];$

$sum[a_, t_] := Block[\{temp = 0, i\},$

$For[i = 1, i <= t, temp = temp + a[[i]]; i++];$

$temp];$

$BranchGenerate[oz, oy, n];$

$AppendTo[LineCounting, Length[LineSet] - sum[LineCounting, 1]];$

$p[a_] := Block[\{temp1, temp2\},$

$If[a == 0, temp1 = 0,$

$For[temp1 = 1; temp2 = 1, temp2 <= a,$

$temp1 = temp1 * (2(n - temp2) + 1); temp2++]$

$];$

$temp1];$

$For[k = 1; i = 1, k < 4,$

$For[r = 1; t = 0, t <= k - 1, r = r + p[t]; t++];$

$For[j = r + 1, j <= r + p[k],$

$BranchGenerate[LineSet[[j]], LineSet[[j + 2(n - k)]], n - k];$

$For[l = j + 1, l <= j + 2(n - k),$

$BranchGenerate[LineSet[[l]], LineSet[[i]], n - k];$

$l++];$

$i++;$

$j = j + 2(n - k) + 1];$

$k++];$

$Show[Table[$

$Graphics3D[Line[\{LineSet[[i, 1]], LineSet[[i, 2]]\}]], \{i, 1, Length[LineSet]\}],$

$AspectRatio-> Automatic, Boxed-> False]$

$Clear[i, j, k, l, r, t];$

$Clear[oy, oz, LineSet, LineCounting, dividenumber, n, \theta];$

$Clear[sum, f, p, LineOnPlane, LineOffPlane, LineNormalize,$
$\quad LineScaling, BranchGenerate];$

Program2 :

$oy = \{\{0, 0, 0\}, \{0, 1, 0\}\};$

$oz = \{\{0, 0, 0\}, \{0, 0, 1\}\};$

$LineSet = \{\{\{0, 0, 0\}, \{0, 0, 1\}\}\};$

$LineCounting = \{1\};$

$\alpha = \pi/3;$

$\theta = 2\pi/3;$

$n = 6;$

$dividenumber = 3.3;$

$LineScaling[a_, t_] := Block[\{temp\},$
$\quad temp = a[[1]] + t * (a[[2]] - a[[1]]);$
$\quad \{a[[1]], temp\}];$

$GetLine[a_, b_, \eta_] :=$
$\quad Block[\{temp, vector, vectora, vectorb, vectorc, length\},$
$\quad\quad length = Norm[a[[2]] - a[[1]]];$
$\quad\quad vectora = Normalize[a[[2]] - a[[1]]];$
$\quad\quad vectorb = b[[2]] - b[[1]];$
$\quad\quad vectorc = Normalize[Cross[vectora, vectorb]];$
$\quad\quad vectorb = Cross[vectorc, vectora];$
$\quad\quad vector = Cos[\alpha] * vectora + Sin[\alpha] * Cos[\eta] * vectorb$
$\quad\quad\quad\quad +Sin[\alpha] * Sin[\eta] * vectorc;$
$\quad\quad temp = a[[1]], a[[1]] + vector;$
$\quad\quad temp = LineScaling[temp, length/dividenumber];$
$\quad\quad temp];$

$BranchGenerate[a_, b_, number_] :=$
$\quad Block[\{i, vector, temp, temp1, temp2\},$
$\quad\quad AppendTo[LineSet, a[[1]] + (number - 1) * (a[[2]] - a[[1]])/number, a[[2]]];$
$\quad\quad For[i = 1, i < number,$
$\quad\quad\quad vector = i * (a[[2]] - a[[1]])/number;$

$temp1 = a[[1]] + vector, a[[2]] + vector;$

$temp2 = \{a[[1]] + vector, a[[1]] + vector + b[[2]] - b[[1]]\};$

$temp = GetLine[temp1, temp2, (-i + 1.5) * \theta];$

$AppendTo[LineSet, temp];$

$i + +]; 0];$

$sum[a_, t_] := Block[\{temp = 0, i\},$

$For[i = 1, i <= t, temp = temp + a[[i]]; i + +];$

$temp];$

$BranchGenerate[oz, oy, n];$

$AppendTo[LineCounting, Length[LineSet] - sum[LineCounting, 1]];$

$total = 3;$

$For[i = 1; k = 1, k <= total, k + +,$

$r = (n^k - 1)/(n - 1);$

$For[j = r + 1, j <= r + n^k, j = j + n,$

$BranchGenerate[LineSet[[j]], LineSet[[j + n - 1]], n];$

$For[l = j + 1, l < j + n, l + +,$

$BranchGenerate[LineSet[[l]], LineSet[[i]], n]$

$];$

$i + +];$

$];$

$Show[Table[Graphics3D[Line[\{LineSet[[i, 1]], LineSet[[i, 2]]\}]],$

$\{i, 1, Length[LineSet]\}], AspectRatio- > Automatic, Boxed- > False]$

$Clear[oy, oz, \alpha, \theta, LineSet, LineCounting, dividenumber];$

$Clear[n, i, j, k, l, p, q, r, total];$

$Clear[sum, LineNormalize, LineScaling, GetLine, BranchGenerate];$

参考文献

[1] 潘金贵著, 分形艺术程序设计,南京大学出版社,1998年3月;

[2] 李尚志,陈发来,张韵华,吴耀华著, 数学实验,高等教育出版社,第二版,2004年8月;